IN THE CLAIMS

1. (Currently amended) A method for tracing the execution path of a computer program comprising at least one module including a plurality of instructions, at least one of said instructions being a branch instruction, the method comprising the steps of:

identifying each branch instruction;

evaluating each branch instruction to be one of true and false; and

responsive to an evaluation of true, pushing at least one unique identifier into a predefined area of storage, wherein said at least one unique identifier itself uniquely identifies ~~corresponds to~~ a set of instructions executed as a result of said evaluation of true.

2. (Previously presented) The method of claim 1, further comprising the step of:

providing said predefined area of storage with volatile memory.

3. (Previously presented) The method of claim 1, further comprising the step of:

providing said predefined area of storage with non-volatile memory.

4. (Previously presented) The method of claim 1, further comprising the step of:

outputting the contents of said storage area to a file at a predetermined point in time.

5. (Previously presented) The method of claim 4, further comprising the step of:

outputting trace information to said file upon exit from said at least one module.

6. (Previously presented) The method of claim 5, further comprising the step of outputting the contents of said storage area at the same time as said exit trace information.

7. (Previously presented) The method of claim 4, wherein the step of outputting the contents of said storage area further comprises the step of:

determining whether said storage area is full; and

responsive to a positive determination, outputting said contents to said file.

8. (Previously presented) The method of claim 4, wherein the step of outputting the contents of said storage area further comprises the step of:

determining whether a failure has occurred within said program; and

responsive to a positive determination, outputting said contents to said file.

9. (Previously presented) The method of claim 4, wherein the step of pushing a unique identifier into a predefined area of storage further comprises the steps of:

determining whether said predefined area of storage is full; and

overwriting the first unique identifier in said storage area.

10. (Previously presented) The method of claim 9, further comprising the step of:

writing the position of the most recent unique identifier to be written out to said storage area to said storage area.

11. (Previously presented) The method of claim 10, further comprising the step of using said position to determine the number of unique identifiers that have been overwritten prior to being written out to said file.

12. (Previously presented) The method of claim 11, further comprising the step of:

responsive to determining that a large number of unique identifiers have been overwritten, increasing the size of said predefined area of storage.

13. (Currently amended) An apparatus for tracing the execution path of a computer program comprising at least one module including a plurality of instructions, at least one of said instructions being a branch instruction, said apparatus comprising:

an identifier for identifying each branch instruction;

an evaluator for evaluating each branch instruction to be one of true and false; and

a pusher, responsive to an evaluation of true, for pushing at least one unique identifier into a predefined area of storage, wherein said at least one unique identifier itself uniquely identifies ~~corresponds to~~ a set of ~~the~~ instructions executed as a result of said evaluation of true.

14. (Currently amended) A method for instrumenting a computer program comprising at least one module including a plurality of instructions, at least one of said instructions being a branch instruction, each branch instruction being evaluated to be one of true and false at run-time, with at least one signature instruction for indicating the execution path of said program at run-time, the method comprising the steps of:

identifying each branch instruction;

identifying ~~the~~ at least one set of instructions associated with an evaluation of true at run-time;

instrumenting said at least one set of instructions associated with an evaluation of true with at least one signature instruction, wherein said at least one signature instruction causes at least one unique identifier to be pushed into a predefined area of storage upon execution of said true instruction at run-time, and wherein said at least one unique identifier itself uniquely identifies ~~corresponds to~~ said at least one set of instructions associated with an evaluation of true.

15. (Currently amended) A compiler for instrumenting a computer program comprising at least one module including a plurality of instructions, at least one of said instructions being a branch instruction, each branch instruction being evaluated to be one of true and false at runtime, with at least one signature instruction for indicating the execution path of said program at run-time, said compiler comprising:

a first identifier for identifying each branch Instruction;

a second identifier for identifying the instructions associated with an evaluation of true at run-time;

a pusher for instrumenting said instructions associated with an evaluation of true with a signature instruction, wherein said signature instruction causes a unique identifier to be pushed into a predefined area of storage upon execution of said true instructions at run-time, and wherein said at least one unique identifier itself uniquely identifies ~~corresponds to~~ said at least one instructions associated with an evaluation of true.